УДК 004.65

Пискунов А.Г.

[1] *канд.физ.-мат. наук, доцент кафедри прикладной математики, НАУ*
[2]

e-mail: agp1@sql.ru

SQLite КАК ХРАНИЛИЩЕ ПРОСТРАНСТВЕННЫХ ДАННЫХ

Описывается приложение, предназначенное для визуализации цифровых карт. Карты при этом хранятся в БД под управлением SQLite, само приложение написано на языке C#.

Ключевые слова: *SQLite, реляционная база данных, пространственные данные.*

Введение

Работа посвящена исследованию возможностей встроенной реляционной базы данных SQLite для хранения пространственных данных, а, именно, различных видео- и фотоматериалов с учетом расположения как самих кадров, так и объектов, обнаруженных на них, на поверхности Земли.

Постановка задачи

Требуется оценить время, которое затрачивает приложение для извлечения из БД данных, необходимых для отрисовки цифровой карты, и, собственно, рисования самой карты. Количество записей в таблицах БД не должно превышать миллион.

Основная часть

В связи с опытом использования реляционной базы данных (а именно Paradox 3.5) для хранения цифровых географических карт см. работы [1–6] было решено использовать бесплатную распространяемую с открытым кодом систему управлениями реляционных баз данных - SQLite [14]. С целью исследования возможностей выбранной СУБД цифровые карты Украины (масштаба 1:500000, содержащую 25000 объектов и полмиллиона точек) и Киевской области (масштаба 1:200000, содержащую

65000 объектов и около миллиона точек) были переконвертированы в базы данных SQLite. Внутрення схема и схема реляционных связей осталась практически такая же как в картографической базе данных (далее - КБД). Электронную версию статьи [6] с описанием КБД на основе СУБД Paradox 3.5 можно посмотреть по адресу [7]. Небольшие изменения внутренней схемы связаны с:

- введением суррогатных ключей, это поля id;
- дальнейшей нормализацией, а именно была удалена таблица $Decoder$, а вместо неё были добавлены две таблицы $Sign$ (характеристики географических объектов) и $Code$ (коды для некоторых харатеристик географических объектов - у таких характеристик в поле $type$ имеет значение S);
- использованием табицы $SgmStr$ для хранения точек сегментов, вместо поля типа blob (Binary Large OBject - Большой Двоичный Объект);
- введением во внутреннюю схему двух индексов, которые будут обсуждаться ниже.

Текущая версия схемы реляционных связей представлена на рис. 1.

Как и в [2, 6] объект (запись в таблице $Object$) имеет тип локализации (точка, линия, многоугольник), код из классификатора [13], несколько характеристик соответственно коду объекта (таблица $ObjSign$), равно как и несколько ломаных, называемых сегментами, которые хранятся в таблице $Segment$. Точки сегментов хранятся в таблице $SgmStr$. Для уменьшения времени выполнения запроса к КБД и сегментам, и объектам дополнительно заранее был высчитан минимальный прямоугольник со сторонами паралелльными осям координат, содержащий все точки образующие сегмент и/или объект. Функция, ставящая в соответствие сегменту или объекту такой прямоугольник, далее будет обозначаться символом box, сам такой прямоугольник будет называться рамка. Далее, через Q обозначим прямоугольник, содержащий точки объектов, которые должны быть отрисованы на экране в текущем запросе.

Версия редактора электронных карт, описанная в [5], сначала находила множество записей из таблицы $Object$:

$$O = \{o | box(o) \cap Q \neq \emptyset \}$$

для объектов, рамки которых имеют непустое пересечение с прямоугольнием запроса. Затем, по полученному множеству записей об объектах, строила множество записей из таблицы $Segment$ (с запоминанием

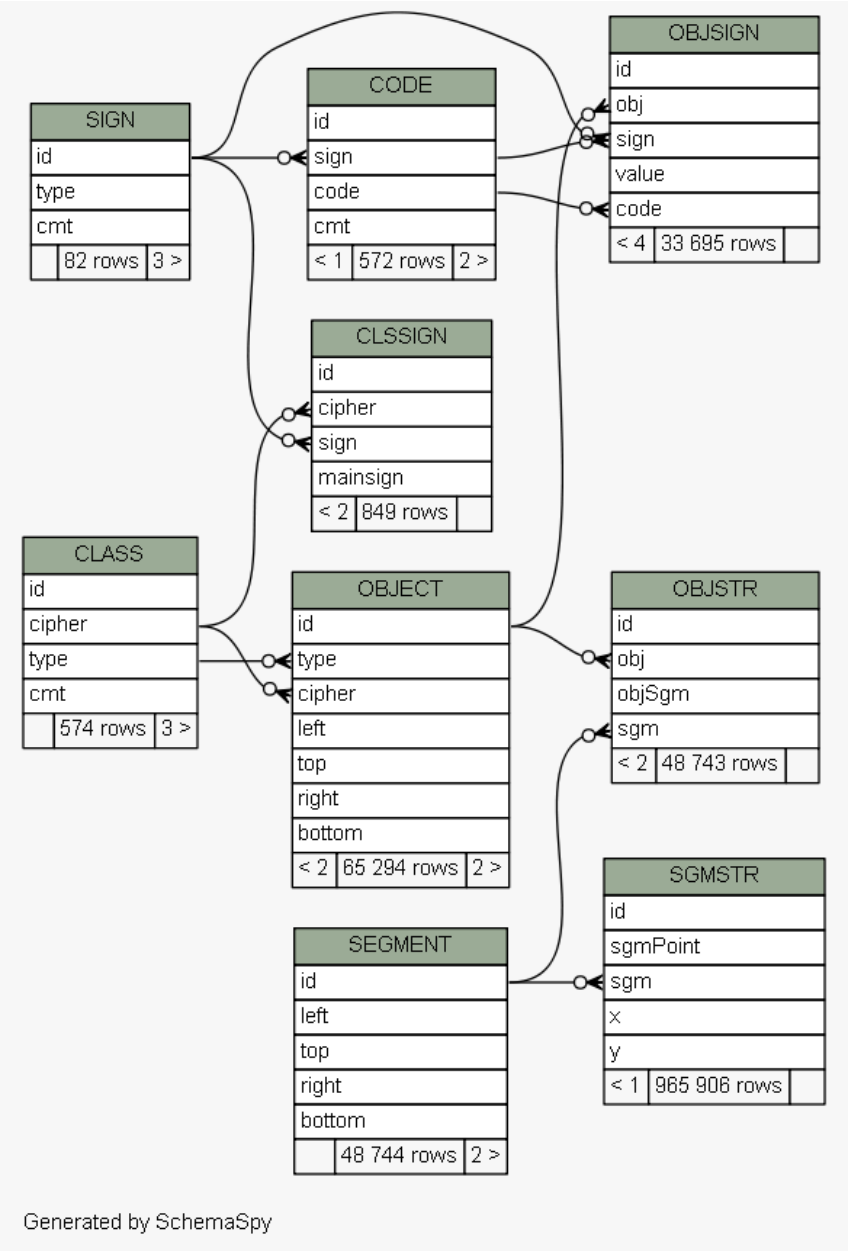

Рис. 1: Схема реляционных связей

способа рисования данного сегмента) пересечение рамок которых с прямоугольником запроса тоже оказывалось не пустым. Далее, для каждого сегмента из поля *sgmStr* таблицы *Segment* извлекались координаты его точек и сегмент отрисовывался на экране.

Текущая версия обозревателя КБД, написанная на языке C# с подключением Ado-провайдера SQLite [10], иначе выполняет поиск требуемых объектов для рисования.

Во-первых, в момент события загрузка окна (грубо говоря, в момент запуска приложения) один раз выполняется запрос:

```
select s.id, s.left, s.top, s.right, s.bottom, o.cipher
```

```
from segment s, objstr os, object o
where   s.id  = os.sgm
  and  os.obj = o.id
```

который создает список S всех номеров сегментов из КБД с географическим кодом (см. [13]) того объекта, в который входит сегмент. В дальнейшем код определяет способ отрисовки данного сегмента. Собственно время выполнения этого первого запроса не превышает двух секунд.

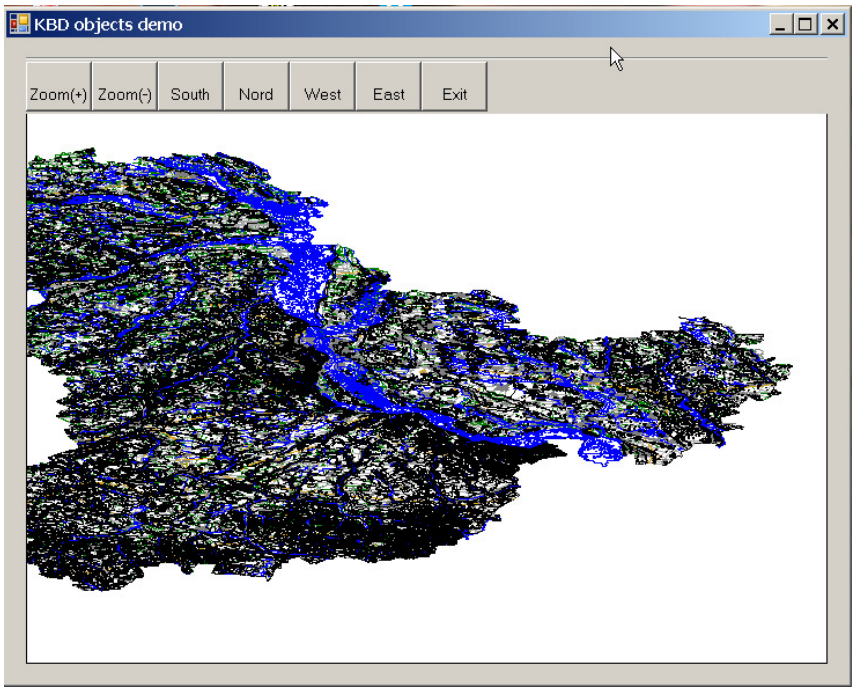

Рис. 2: Отрисовка всей карты Киевской области

Во-вторых, при каждом изменении прямоугольника Q, для сегментов из S, пересечение рамок которых с прямоугольником запроса не пусто, выбираются их точки:

```
select x, y
from SgmStr
where sgm = @sgm
order by sgmPoint
```

и полностью строится хеш-таблица T из пар (номер_сегмента, список_его_точек).

По окончании построения таблицы T выполняется отрисовка карты. На рис. 2 представлено окно с прорисованной картой Киевской области.

Далее, так как для поиска точек при выполнении запроса в редакторе электронных карт, программа последовательно выбирала записи из таблицы *Object*, затем - с использованием первичных ключей - рамок

и точек сегментов из таблицы *Segment*, то нужда в использовании дополнительных индексов отсутствовала. Текущий обозреватель карт для нормального выполнения обеих запросов требует введение двух индексов:

- `create index objsgm_sgmi on ObjStr (sgm);`
  - ускоряет выполнение первого запроса для построении списка S;
- `create index sgmenti on SgmStr (sgm, sgmPoint);`
  - ускоряет выполнение второго запроса для построения хеш-таблицы T.

Так как построение хеш-таблицы каждый раз выполняется полностью, то одним из способов ускорения времени реакции является создание специального кэша на базе R-дерева [11], который не будет перестраиваться целиком при каждом изменении запроса Q и, возможно, фоновой подкачки новых сегментов, как это предлагалось делать в [8].

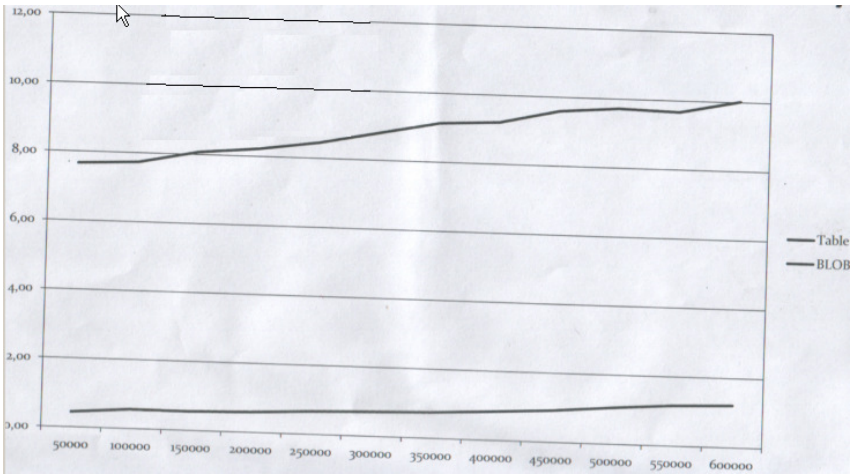

Рис. 3: Время отработки запроса к КБД

Обсуждаемый обозреватель КБД на весьма средних компьютерах производства 2005-2007 годов выполняет выполнять выборку данных для полной отрисовки карты Украины за 15 секунд, а выборку данных для полной отрисовки карты Киевской области за 30 секунд. Такое время реакции программы трактуется как вполне приемлемое. Кроме того, были поставлены эксперименты по переносу точек сегментов из отдельной таблицы в поле типа blob таблицы *Segment*. Это приводило к существенному (5-8 раз) повышению производительности обозревателя на рассматриваемом количестве точек (до миллиона). Причем скорость выполнения запроса в случае с таблицей *SgmStr*, и в случае без неё падала линейно в зависимости от количества выбираемых точек. Результаты экспериментов собраны на графике 3 , где на оси X отмечается количество точек,

на оси Y - время выполнения запроса в секундах.

На практике предполагается брать растровые карты из доступных источников. К таким относится, например, проект OpenStreetMap см. [12]. При этом, в КБД будет хранится только координаты снимков и объектов, обнаруженных на этих снимках. Предполагается, что общее число записей для хранения требуемой информации будет меньше, чем в двух рассмотренны картах. Вопрос, совмещения растровых карт проекта OpenStreetMap и пространственных данных из КБД прорабатывался в [9]. Таким образом, производительность обозревателя карт на основе СУБД SQLite признана вполне удовлетворительной для предполагаемого использования.

Список используемых источников

[1] *Пискунов А.Г.* Концептуальная схема системы управления электронными картами / А.Г.Пискунов, В.О.Федченко // Планирование экономического развития регионов с учетом их экологической безопастности: тезисы докладов семинара. – Севастополь, 1993.

[2] *Пискунов А.Г.* Внутренняя схема системы управления электронными картами / А.Г.Пискунов, В.О.Федченко //
Контроль и управление в технических системах: тезисы докладов конференции стран СНГ. – Винница, 1993.

[3] *Пискунов А.Г.* Архитектура редактора электронных карт / А.Г.Пискунов, В.О.Федченко, В.Г.Малышко // Автоматика-94: тезисы 1-й Украинской конференции по автоматическому управлению. - Киев, 1994.

[4] *Пискунов А.Г.* Система преобразования форматов карт / А.Г. Пискунов, В.Д. Барановский, А.А.Мокринцев // Автоматика-95 : тезисы 2-й Украинской конференции по автоматическому управлению. - Львов, 1995.

[5] *Пискунов А.Г.* Повышение производительности системы управления электронными картами / А.Г. Пискунов, В.О. Федченко, М.В. Николаев // Автоматика-95 : тезисы 2-й Украинской конференции по автоматическому управлению. - Львов, 1995.

[6] *Пискунов А.Г.* Использование реляционной модели при создании картографической базы данных / А.Г. Пискунов, В.О. Федченко, М.В. Николаев // Применение компьютерных технологий в решении задач народного хозяйства : сборник научных трудов института кибернетики НАНУ. – Киев, 1996.

[7] *Пискунов А.Г.* Использование реляционной модели при создании картографической базы данных [электронный ресурс] / А.Г. Пискунов, В.О. Федченко, М.В. Николаев. - Режим доступа: `http://agp1.hx0.ru/kbd.html`.

[8] *Сорокопуд В.І.* Оптимізація запитів до інформаційних систем обробки геопросторових даних / В.І. Сорокопуд //ПОЛІТ.Сучасні проблеми науки. Інформаційно-діагностичні системи: тези XIV міжнародної науково-практичної конференції молодих учених і студентів. - Київ: НАУ, 2015. - 147 с.

[9] *Соснєв Л.М.* Використання відкритих сервісів геопросторових даних в інформаційних системах на прикладі OpenSreetMap /Л.М. Соснєв, А.О.Колесник, В.Д. Зівакін // ПОЛІТ.Сучасні проблеми науки. Інформаційно-діагностичні системи: тези XIV міжнародної науково-практичної конференції молодих учених і студентів. - Київ: НАУ, 2015. - 147 с.

[10] Ado-провайдер System.Data.SQLite [электронный ресурс]. - Режим доступа: `http://system.data.sqlite.org/index.html/doc/trunk/www/downloads.wiki`.

[11] *Гуттман Антонин.* R-дерево [электронный ресурс] / Антонин Гуттман. - Режим доступа: `http://iipo.tu-bryansk.ru/fileadmin/user_upload/trubakov/book/3_2_1_R_Tree.pdf`.

[12] *Стив Кост* Некоммерческий картографический проект OSM [электронный ресурс] / Стив Кост. - Режим доступа: `http://www.openstreetmap.org`.

[13] Классификатор графических изображений / Редакционно - издательский отдел, Военно - топографическое управление. - М: ЕСКиК-КиИ, 1988.

[14] Встраиваемая система управления базами данных SQLite [электронный ресурс]. Режим доступа : `http://sqlite.org`.